Amendments to the Claims:

This listing of Claims will replace all prior versions and listings of Claims in the Application:

Listing of Claims:

---

a2

1. (currently amended) A method comprising:

determining a set of arguments for an outsourced computation;

classifying, with a first computer, the outsourced computation into one of a number of computation types;

selecting, with the first computer, one or more disguising operations from a predetermined set of disguising operations based on said classifying;

~~preparing a group of disguised arguments corresponding to the set of~~ performing the one or more selected disguising operations on the actual arguments with [[a]] the first computer to provide disguised arguments;

outputting the disguised arguments from the first computer for performance of the outsourced computation; and

receiving a result of the outsourced computation performed with the disguised arguments.

2. (original) The method of claim 1, further comprising computing an actual answer from the result after said receiving.

3. (cancelled)

4. (currently amended) The method of claim [[3]] 1, wherein the computation types ~~include~~ comprise one or more computation types selected from the group consisting of matrix multiplication computations, matrix inversion computations, and convolution computations.

a2

5. (currently amended) The method of claim [[4]] 1, wherein the computation types ~~further include solution of~~ comprise one or more computation types selected from the group consisting of computations for solving a system of linear equations, computations for solving one or more differential equations, quadrature computations, image edge detection computations, character string pattern matching computations, and sorting computations.

6. (original) The method of claim 1, further comprising:

receiving the disguised arguments at a second computer remotely located relative to the first computer;

performing the outsourced computation with the second computer; and

sending the result from the second computer to the first computer, the result being in a disguised form relative to an answer obtained by submitting the actual arguments to the outsourced computation.

7. (currently amended) The method of claim 1, wherein said preparing includes generating a plurality of random numbers, the random numbers each being generated by one of a number of random number generation techniques, the techniques each ~~including~~ comprising a different distribution parameter.

a2 8. (currently amended) The method of claim 7, wherein said preparing further ~~includes~~ comprises defining a number of disguise functions with one or more of the random numbers.

9. (currently amended) The method of claim 1, wherein said preparing ~~includes~~ comprises modifying a linear operator.

10. (currently amended) The method of claim 1, wherein said preparing ~~includes~~ comprises altering a dimension corresponding to the actual arguments to provide the disguised arguments.

11. (currently amended) The method of claim 10, wherein said altering ~~includes~~ comprises expanding the dimension.

12. (currently amended) The method of claim 1, wherein said preparing ~~includes~~ comprises performing a function substitution in accordance with at least one mathematical identity.

13. (cancelled)

14. (cancelled)

15. (cancelled)

16. (cancelled)

17. (cancelled)

a2

18. (currently amended) A system comprising:

a computer operable to define a set of actual arguments for an outsourced computation, said computer being programmed to classify said computation into at least one of a plurality of computation types, said computer being programmed to determine a group of disguised arguments from the set of actual arguments, said disguised arguments hiding one or more characteristics of the set of actual arguments;

an output device responsive to said computer to output the disguised arguments for remote performance of said outsourced computation; and

an input device to receive a result of said outsourced computation performed with said disguised arguments, [[; and]] wherein said computer is responsive to said input device to determine a desired answer from said result.

19. (currently amended) The system of claim 18, wherein ~~said computer is further programmed to classify said outsourced~~ the computation ~~as being one of a number of types, said~~

~~types including at least one~~ types comprise one or more computation types selected from the group consisting of matrix multiplication computations, matrix inversion computations, and convolution computations.

20. (original) The system of claim 18, further comprising a computing center, said computing center being programmed to perform said outsourced computation with said disguised arguments.

21. (original) The system of claim 18, wherein said computer includes a memory, a library of disguise operations being stored in said memory, said computer programming referencing said library to generate said disguised arguments.

22. (original) The system of claim 21, wherein said disguise operations correspond to at least one of the group consisting of random object generation, argument dimension modification, mathematical identity substitution, and disguise function generation.

23. (original) The system of claim 18, wherein said computer includes instructions to generate a cubic spline to provide a disguise for said actual arguments.

24. (cancelled)

25. (cancelled)

26. (cancelled)

27. (cancelled)

28. (currently amended) An apparatus, comprising: a computer readable medium, said medium defining computer programming instructions to hide a group of actual arguments for a computation to be outsourced, said programming instructions being operable to classify said computation into at least one of a plurality of computation types, said instructions being operable to generate a group of disguised arguments corresponding to said actual arguments, said disguised arguments being generated to provide a disguised result when provided for said computation, an actual answer being recoverable from said disguised result in accordance with said instructions, said actual answer being returned by said computation when said computation is provided said actual arguments.

29. (original) The apparatus of claim 28, further including a computer responsive to said programming instructions.

30. (currently amended) The apparatus of claim 28, wherein said ~~programming instructions are executable to classify said~~ computation ~~into at least one of a plurality of~~ types comprise one or more computation types~~, said computation types including~~ selected from the group consisting

of matrix multiplication computations, matrix inversion computations, and convolution computations.

31. (original) The apparatus of claim 28, wherein said programming instructions further define a library of disguise operations, said disguise operations corresponding to at least one of the group consisting of random object generation, dimension modification, and mathematical identity substitution.

a2

32. (original) The apparatus of claim 28, wherein said programming instructions define a routine to generate a cubic spline to provide at least one disguise function.

33. (original) The apparatus of claim 28, wherein said programming instructions define a routine to provide a random function space to provide one or more disguise functions.

34. (new) A system comprising:

a computer operable to define a set of actual arguments for an outsourced computation, said computer being programmed to determine a group of disguised arguments from said set of actual arguments, said computer comprising instructions to generate a cubic spline to provide a disguise for said actual arguments, said disguised arguments hiding one or more characteristics of said set of actual arguments;

an output device responsive to said computer to output said disguised arguments for remote performance of said outsourced computation;

an input device to receive a result of said outsourced computation performed with said disguised arguments; and

wherein said computer is responsive to said input device to determine a desired answer from said result.

a2

35. (new) An apparatus comprising:

a computer readable medium, said medium comprising computer programming instructions to hide a group of actual arguments for a computation to be outsourced, said programming instructions being operable to generate a group of disguised arguments corresponding to said actual arguments, said programming instructions comprising a routine to generate a cubic spline to provide at least one disguise function, said disguised arguments being generated to provide a disguised result when provided for said computation, an actual answer being recoverable from said disguised result in accordance with said instructions, said actual answer being returned by said computation when said computation is provided said actual arguments.

36. (new) The method of claim 1, wherein said preparing comprises performing a function substitution in accordance with at least one expansion of unity.

37. (new) A method for outsourcing a matrix multiplication, the method comprising the steps of:

providing, in a memory of a first computer, a first actual matrix M1 comprising a first plurality of actual arguments, and a second actual matrix M2 comprising a second plurality of actual arguments;

preparing, in said memory of said first computer, at least two disguising matrices, each said disguising matrix being a sparse matrix comprising at least one non-zero disguising argument, wherein each said at least one non-zero disguising argument comprises a random number;

a2

computing, with said first computer, a first disguised matrix X using said first actual matrix and at least one said disguising matrix, said first disguised matrix X comprising a first plurality of disguised arguments; and

computing, with said first computer, a second disguised matrix Y using said second actual matrix and at least one disguising matrix, said second disguised matrix Y comprising a second plurality of disguised arguments.

38. (new) The method of claim 37, further comprising the steps of:

outputting said disguised arguments from said first computer for performance of a matrix multiplication; and

receiving, with said first computer, a result of said matrix multiplication.

39. (new) The method of claim 38, further comprising, after the receiving step, the step of:

computing an actual answer from said result.

40. (new) The method of claim 38, further comprising the steps of:

receiving said disguised arguments at a second computer;

performing said matrix multiplication with said second computer; and

sending said result from said second computer to said first computer, said result being in a disguised form relative to an answer obtained by submitting said actual arguments to said second computer.

a2

41. (new) The method of claim 37, wherein said first actual matrix M1 is a non-square matrix.

42. (new) The method of claim 37, wherein said second actual matrix M2 is a non-square matrix.

43. (new) The method of claim 37, further comprising, before the step of computing said first disguised matrix X, the step of:

changing a dimension of said first actual matrix M1.

44. (new) The method of claim 37, further comprising, before the step of computing said second disguised matrix Y, the step of:

changing a dimension of said second actual matrix M2.

45. (new) The method of claim 37, further comprising the steps of:

preparing, in said memory of said first computer, a first random matrix S_1 comprising a first plurality of random arguments, and a second random matrix S_2 comprising a second plurality of random arguments;

generating, with said first computer, a first random number β, a second random number γ, a third random number β', and a fourth random number γ';

a2

computing, with a second computer, matrices W, U, and U' as follows:

$$W = (X + S_1)(Y + S_2),$$

$$U = (\beta X - \gamma S_1)(\beta Y - \gamma S_2), \text{ and}$$

$$U' = (\beta' X - \gamma' S_1)(\beta' Y - \gamma' S_2);$$

computing, with said first computer, matrices V and V' as follows:

$$V = (\beta + \gamma)^{-1} (U + \beta\gamma W), \text{ and}$$

$$V' = (\beta' + \gamma')^{-1} (U' + \beta'\gamma' W);$$

computing, with said second computer, a matrix Z as follows:

$$Z = (\gamma'\beta - \gamma\beta')^{-1} (\gamma' V - \gamma V'); \text{ and}$$

deriving, with said first computer, a multiplicative product of said first actual matrix M1 and said second actual matrix M2 from said matrix Z.

46. (new) The method of claim 45, wherein said first actual matrix M1 is a non-square matrix.

47. (new) The method of claim 45, wherein said second actual matrix M2 is a non-square matrix.

51. (new) The method of claim 50, wherein said matrix Q is determined to be invertible, the method further comprising the steps of:

(f) inverting, with said second computer, said matrix Q to create an inverse matrix Q^{-1};

(g) computing, with said first computer, a matrix T as follows:

$$T = P4 * P2^{-1} * Q^{-1} * P1 * P5^{-1};$$

a2

(h) computing, with said first computer, a matrix R as follows:

$$R = P3 * S * P4^{-1};$$

(i) computing, with said second computer, a matrix Z as follows: $Z = R * T$; and

(j) deriving, with said first computer, an inverse of said matrix M from said matrix Z.

52. (new) The method of claim 50, wherein said matrix Q is determined not to be invertible, the method further comprising the steps of:

(f) computing, with said first computer, a matrix S' as follows: $S' = S_1 * S * S_2$, where S_1 and S_2 are matrices known to be invertible;

(g) determining, with said second computer, whether said matrix S' is invertible; and

(h) if said matrix S' is determined not to be invertible, repeating steps (b)-(e).

48. (new) The method of claim 45, further comprising, before the step of computing said first disguised matrix X, the step of:

changing a dimension of said first actual matrix M1.

49. (new) The method of claim 45, further comprising, before the step of computing said second disguised matrix Y, the step of:

changing a dimension of said second actual matrix M2.

a2

50. (new) A method for outsourcing a matrix inversion computation, the method comprising the steps of:

(a) providing, in a memory of a first computer, a first actual matrix M comprising a first plurality of actual arguments;

(b) providing, in said memory of said first computer, a random matrix S comprising a plurality of random arguments;

(c) generating, with said first computer, a first disguising matrix P1, a second disguising matrix P2, a third disguising matrix P3, a fourth disguising matrix P4, and a fifth disguising matrix P5, wherein each said disguising matrix is a sparse matrix comprising at least one non-zero disguising argument, wherein each said at least one non-zero disguising argument comprises a random number;

(d) computing, with said first computer, a matrix Q as follows:

$$Q = P1 * M * S * P2^{-1}; \text{ and}$$

(e) determining, with a second computer, whether said matrix Q is invertible.

53. (new) The method of claim 50, further comprising, before the step of providing said random matrix S, the step of:

changing a dimension of said first actual matrix M.

54. (new) A method for outsourcing the computation of a solution vector for a system of linear equations of the form Mx=b, wherein M is a matrix having *n* rows and *n* columns wherein *n* is a positive integer, wherein b is a vector of dimension *n*, and wherein x is a solution vector, the method comprising the steps of:

generating, with a first computer, a random matrix B having the same dimensions as matrix M and comprising a plurality of random arguments;

generating, with said first computer, a random integer j such that $1 \leq j \leq n$;

replacing, in a memory of said first computer, a row of said random matrix B corresponding to said random integer j with said vector b;

preparing, in said memory of said first computer, at least two disguising matrices, each said disguising matrix being a sparse matrix comprising at least one non-zero disguising argument, wherein each said at least one non-zero disguising argument comprises a random number;

computing, with said first computer, a matrix C' using said matrix M and at least two of said at least two disguising matrices;

computing, with said first computer, a matrix G' using said random matrix B and at least two of said at least two disguising matrices;

computing, with a second computer, a matrix U as follows: $U = C^{r-1} * G'$;

computing, with said first computer, a matrix X using said matrix U and at least two of said at least two disguising matrices; and

a2

deriving, with said first computer, said solution vector x from said matrix X.

55. (new) A method for outsourcing the computation of a solution vector for a system of linear equations of the form Mx=b, wherein M is a matrix having *n* rows and *n* columns wherein *n* is a positive integer, wherein b is a vector of dimension *n*, and wherein x is a solution vector, the method comprising the steps of:

in a memory of a first computer, embedding matrix M in a larger matrix M';

in said memory of said first computer, embedding vector b in a larger vector b';

computing a solution vector x' that satisfies equation M'x' = b', wherein said computation of said solution vector x' is allocated between said first computer and a second computer with each of said first computer and said second computer performing at least a portion of said computation of said solution vector x'; and

deriving, with said first computer, said solution vector x from said solution vector x'.

56. (new) A method for outsourcing the computation of an estimate for the solution of a quadrature computation of the form

$$\int_a^b f(x)dx,$$

wherein the estimate to be computed must conform to a predetermined level of accuracy, the method comprising the steps of:

a2

creating, in a memory of a first computer, at least seven numbers y_i, wherein i is an integer index variable having a property of $1 \leq i \leq max$, wherein $y_1 = a$ and $y_{max} = b$, and wherein said other numbers y_i satisfy the following properties:

$a < y_i < b$, and

$y_{i-1} < y_i$;

creating, in said memory of said first computer, a number of values v_i, wherein i is an integer index variable having a property of $1 \leq i \leq max$, wherein there are the same number of values v_i as numbers y_i, and wherein said values v_i satisfy the following properties:

$v_{i-1} < v_i$, and

$\min |f(x)| \approx v_1 \leq v_{max} \approx \max |f(x)|$, wherein the operations $\min |f(x)|$ and $\max |f(x)|$ return the minimum and maximum absolute value of function f(x), respectively;

creating, in said memory of said first computer, a cubic spline g(y) with breakpoints comprising said numbers y_i such that $g(y_i) = v_i$;

integrating, with said first computer, cubic spline g(y) from a to b to obtain a value I_1;

computing, with a second computer, a value I_2 using said cubic spline g(y), said function f(x), and a designation of said predetermined level of accuracy; and

computing, with said first computer, said estimate by subtracting said value I_1 from said I_2.

57. (new) A method for outsourcing a convolution computation of two vectors, the method comprising the steps of:

providing, in a memory of a first computer, a first vector M_1, and a second vector M_2, said first vector M_1 and said second vector M_2 being of equivalent size;

creating, in said memory of said first computer, a first random vector S_1 and a second random vector S_2, said first random vector S_1 and second random vector S_2 being the same size as said first vector M_1 and said second vector M_2;

generating, with said first computer, a first random number α, a second random number β, a third random number γ, a fourth random number β', and a fifth random number γ';

computing, with a second computer, convolutions W, U, and U' as follows:

$$W = (\alpha M_1 - S_1) \otimes (\alpha M_2 + S_2),$$

$$U = (\beta M_1 - \gamma S_1) \otimes (\beta M_2 - \gamma S_2), \text{ and}$$

$$U' = (\beta' M_1 - \gamma' S_1) \otimes (\beta' M_2 - \gamma' S_2);$$

computing, with said first computer, vectors V and V' as follows:

$V = (\beta + \alpha\gamma)^{-1} (\alpha U + \beta\gamma W)$, and

$V' = (\beta' + \alpha\gamma')^{-1} (\alpha U' + \beta'\gamma' W)$; and

deriving, with said first computer, a convolution of vectors M1 and M2 utilizing vectors V and V'.

58. (new) The method of claim 57, further comprising, before the step of creating said first random vector S_1 and said second random vector S_2, the step of:

altering, in said memory of said first computer, said size of said first vector M_1 and said second vector M_2.

59. (new) A method for outsourcing the computation of a solution to a linear differential equation $Ly = F(x,y)$ of order N, wherein N is a positive integer, the linear differential equation having boundary conditions $y(x_i) = y_i$, wherein i is an integer index variable and $0 \leq i < N$, the method comprising the steps of:

creating, in a memory of a first computer, a cubic spline g(x) and a function $u(x) = Lg(x)$, wherein Lg(x) is a linear differential equation of order N;

deriving, with a second computer, solution function z(x) for the following set of equations:

$Ly = F(x,y) + u(x)$, and

$y(x_i) = y_i + u(x_i)$;

deriving, with said first computer, a solution to said linear differential equation Ly from said derived solution function z(x).

60. (new) A method for disguising a symbolic mathematical expression, the method comprising the step of:

automatically applying one or more transformation techniques to said symbolic mathematical expression, said one or more transformation techniques selected from the group consisting of disguising constants in said symbolic mathematical expression and replacing variable names in said symbolic mathematical expression.

61. (new) A method for disguising a symbolic mathematical expression, the method comprising the step of:

automatically applying one or more transformation techniques to said symbolic mathematical expression, said one or more transformation techniques selected from the group consisting of applying at least one mathematical identity function to said symbolic mathematical expression and applying at least one expansion of unity to said symbolic mathematical expression.

62. (new) A method for outsourcing a computation for detecting edges of an image, wherein the image is represented by an nxn array of pixel values p(x,y) between 0 and 100,000 on a unit square $0 \leq x,y \leq 1$, wherein n is a positive integer, the method comprising the steps of:

creating, in a memory of a first computer, at least ten ordered number pairs x_i,y_i, wherein i is an integer index variable having a property of $1 \leq i \leq max$, wherein $x_1,y_1 = 0$,

wherein $x_{max},y_{max}= 1$, and wherein the remaining ordered number pairs x_i,y_i satisfy the following properties:

$0 < x_i,y_i < 1$, and

$x_{i-1},y_{i-1} < x_i,y_i$;

creating, in a memory of a first computer, a plurality of random values $v_{i,j}$ such that $0 \leq v_{i,j} \leq 50{,}000$;

creating, in a memory of a first computer, four number pairs a_k,b_k, wherein k is an integer index variable having a property of $1 \leq k \leq 4$, each said number pair a_k,b_k comprising positive random numbers such that $a_1 = \min(a_k)$, $a_4 = \max(a_k)$, $b_1 = \min(b_k)$, and $b_4 = \max(b_k)$;

creating, in a memory of said first computer, a bi-cubic spline s(x,y), such that $s(x_i,y_i) = v_{i,j}$;

determining, with said first computer, a linear change of coordinates from (x,y) coordinates to (u,v) coordinates that maps said unit square into a rectangle with vertices (a_k,b_k);

converting, with said first computer, said pixel values p(x,y) to pixel values p(u,v);

converting, with said first computer, said bi-cubic spline s(x,y) to a bi-cubic spline s(u,v);

computing, with said second computer, an image e(u,v) using said pixel values p(u,v) and said bi-cubic spline s(u,v); and

computing, with said first computer, edges of said image e(u,v).

63. (new) A method for outsourcing a template matching computation for image analysis, the template matching computation of a score matrix $C_{I,P}$, the score matrix $C_{I,P}$ comprising an approximation of whether an image object P of size *n*x*n* appears in a larger image I of size *N*x*N*, wherein *n* and *N* are positive integers, the method comprising the steps of:

generating, with a first computer, a random matrix S1 of size *N*x*N*, and a random matrix S2 of size *n*x*n*;

generating, with said first computer, first random number α, second random number β, third random number γ, fourth random number β', and fifth random number γ';

computing, with said first computer the following matrices:

$\alpha I + S1$,

$\alpha P + S2$,

$\beta I - \gamma S1$,

$\beta P - \gamma S2$,

$\beta' I - \gamma' S1$, and

$\beta' P - \gamma' S2$;

computing, with a second computer, matrices W, U, and U' as follows:

$W = C_{(\alpha I + S1),(\alpha P+S2)}$,

$U = C_{(\beta I - \gamma S1)(\beta P - \gamma S2)}$, and

$U' = C_{(\beta' I - \gamma' S1),(\beta' P-\gamma' S2)}$;

computing, with said first computer, matrices V and V' as follows:

$$V = (\beta + \alpha\gamma)^{-1} (\alpha U + \beta\gamma W), \text{ and}$$

$$V' = (\beta + \alpha\gamma)^{-1} (\alpha U' + \beta'\gamma' W); \text{ and}$$

deriving, with said first computer, said score matrix $C_{I,P}$ from said matrices V and V'.

64. (new) A method for securely outsourcing the sorting of a sequence of *n* numbers $E = \{e_1,..., e_n\}$, wherein *n* is a positive integer, the method comprising the steps of:

selecting, with a first computer, a strictly increasing function f();

generating, with said first computer, a random sorted sequence $\Lambda = \{\lambda_1,...,\lambda_n\}$ of *n* numbers;

computing, with said first computer, sequence E' = f(E) and sequence $\Lambda' = f(\Lambda)$, where f(E) is said sequence obtained from E by replacing every element e_i by $f(e_i)$;

computing, with said first computer, set W as follows: $W = E' \cup \Lambda'$;

randomly permuting set W with said first computer;

sorting, with a second computer, randomly permuted set W to derive sorted set W'; and

deriving, with said first computer, sorted sequence E from sorted set W'.

65. (new) A method for secure outsourcing of a text string pattern matching computation, wherein T is a text string of length *N*, wherein *N* is a positive integer, said text string T

comprising N text symbols, and wherein P is a text pattern of length n, wherein n is a positive integer that is smaller than N, said text pattern P comprising n text symbols, and wherein alphabet A comprises the plurality of possible text symbols that could appear in text string T or text pattern P, the method comprising the steps of:

a2

(a) selecting a text symbol from alphabet A;

(b) replacing, with a first computer, each instance of said selected text symbol in said text string T with the number 1, and replacing each other text symbol in text string T with the number 0, the resultant text string being designated T_x;

(c) replacing, with a first computer, each instance of said selected text symbol in said text pattern P with the number 1, and replacing each other text symbol in text pattern P with the number 0, the resultant text pattern being designated P_x;

(d) augmenting, with a first computer, said text pattern P_x into a text string P' of length by adding zeros thereto;

(e) computing, with a second computer, a value D_x as follows:

$$D_x(i) = \sum_{k=0}^{n-1} T_x(i+k)P'(k), \qquad 0 \le i \le N-n;$$

(f) repeating steps (a)-(e) until all text symbols from said alphabet A have been selected one time; and

(g) computing, with said first computer, a score matrix $C_{T,P}$ using all said values D_x.

66. (new) A computer comprising:

computer circuitry configured to define a first actual matrix M1 comprising a first plurality of actual arguments, and to define a second actual matrix M2 comprising a second plurality of actual arguments;

a^2

computer circuitry configured to prepare at least two disguising matrices, each said disguising matrix being a sparse matrix comprising at least one non-zero disguising argument, wherein each said at least one non-zero disguising argument comprises a random number;

computer circuitry configured to compute a first disguised matrix X using said first actual matrix and at least one said disguising matrix, said first disguised matrix X comprising a first plurality of disguised arguments; and

computer circuitry configured to compute a second disguised matrix Y using said second actual matrix and at least one disguising matrix, said second disguised matrix Y comprising a second plurality of disguised arguments.

67. (new) The computer of claim 66, further comprising:

computer circuitry configured to output said first plurality of disguised arguments and said second plurality of disguised arguments for performance of a matrix multiplication;

computer circuitry configured to receive a result of said matrix multiplication; and

computer circuitry configured to compute an actual answer from said result.

68. (new) The computer of claim 66, further comprising:

computer circuitry configured to prepare a first random matrix S_1 comprising a first plurality of random arguments, and to prepare a second random matrix S_2 comprising a second plurality of random arguments;

a2

computer circuitry configured to generate a first random number β, a second random number γ, a third random number β', and a fourth random number γ';

computer circuitry configured to compute a first set of disguised arguments from the following matrix operations:

$$(X + S_1),$$

$$(Y + S_2),$$

$$(\beta X - \gamma S_1),$$

$$(\beta Y - \gamma S_2),$$

$$(\beta' X - \gamma' S_1), \text{ and}$$

$$(\beta' Y - \gamma' S_2);$$

computer circuitry configured to output said first set of disguised arguments;

computer circuitry configured to receive a matrix U, a matrix U', and a matrix W computed from said first set of disguised arguments;

computer circuitry configured to compute a second set of disguised arguments comprising matrices V and V' as follows:

$$V = (\beta + \gamma)^{-1} (U + \beta\gamma W), \text{ and}$$

$$V' = (\beta' + \gamma')^{-1} (U' + \beta'\gamma' W);$$

computer circuitry configured to output said second set of disguised arguments;

computer circuitry configured to receive a matrix Z computed from said second set of disguised arguments; and

computer circuitry configured to derive a multiplicative product of said first actual matrix M1 and said second actual matrix M2 from said matrix Z.

a2

69. (new) A computer comprising:

computer circuitry configured to define a first actual matrix M comprising a first plurality of actual arguments;

computer circuitry configured to prepare a random matrix S comprising a plurality of random arguments;

computer circuitry configured to generate a first disguising matrix P1, a second disguising matrix P2, a third disguising matrix P3, a fourth disguising matrix P4, and a fifth disguising matrix P5, wherein each said disguising matrix is a sparse matrix comprising at least one non-zero disguising argument, wherein each said at least one non-zero disguising argument comprises a random number; and

computer circuitry configured to compute matrix Q as follows:

$$Q = P1 * M * S * P2^{-1}.$$

70. (new) The computer of claim 69, further comprising:

computer circuitry configured to receive an inverse matrix Q^{-1} of said matrix Q;

computer circuitry configured to compute a matrix T as follows:

$T = P4 * P2^{-1} * Q^{-1} * P1 * P5^{-1}$;

computer circuitry configured to compute, a matrix R as follows:

$R = P3 * S * P4^{-1}$;

computer circuitry configured to output said matrices T and R;

computer circuitry configured to receive a matrix Z computed from said matrices T and R; and

computer circuitry configured to derive an inverse of said matrix M from said matrix Z.

71. (new) The computer of claim 69, further comprising:

computer circuitry configured to compute a matrix S' as follows: $S' = S_1 * S * S_2$, wherein S_1 and S_2 are matrices known to be invertible; and

computer circuitry configured to output said matrix S'.

72. (new) A computer for use in the computation of a solution vector for a system of linear equations of the form Mx=b, wherein M is a matrix having *n* rows and *n* columns wherein *n* is a positive integer, wherein b is a vector of dimension *n*, and wherein x is a solution vector, the computer comprising:

computer circuitry configured to generate a random matrix B having the same dimensions as matrix M and comprising a plurality of random arguments;

computer circuitry configured to generate a random integer j such that $1 \leq j \leq n$;

computer circuitry configured to replace a row of said random matrix B corresponding to said random integer j with said vector b;

computer circuitry configured to prepare at least two disguising matrices, each said disguising matrix being a sparse matrix comprising at least one non-zero disguising argument, wherein each said at least one non-zero disguising argument comprises a random number;

computer circuitry configured to compute a matrix C' using said matrix M and at least two of said at least two disguising matrices;

computer circuitry configured to compute a matrix G' using said random matrix B and at least two of said at least two disguising matrices;

computer circuitry configured to output said matrices C' and G';

computer circuitry configured to receive a matrix U computed from said matrix G' and the inverse of said matrix C';

computer circuitry configured to compute a matrix X using said matrix U and at least two of said at least two disguising matrices; and

computer circuitry configured to derive said solution vector x from said matrix X.

73. (new) A computer for use in the computation of a solution vector for a system of linear equations of the form Mx=b, wherein M is a matrix having *n* rows and *n* columns wherein *n* is a

positive integer, wherein b is a vector of dimension *n*, and wherein x is a solution vector, the computer comprising:

computer circuitry configured to embed said matrix M in a matrix M', said matrix M' being larger than said matrix M;

computer circuitry configured to embed said vector b in a vector b', said vector b' being larger than said vector b;

computer circuitry configured to outsource to another computer at least a portion of the computation of a solution vector x' that satisfies equation M'x' = b'; and

computer circuitry configured to derive said solution vector x from said solution vector x'.

74. (new) A computer for use in the computation of an estimate for the solution of a quadrature computation of the form

$$\int_a^b f(x)dx,$$

wherein the estimate to be computed must conform to a predetermined level of accuracy, the method comprising the steps of:

computer circuitry configured to generate at least seven numbers y_i, wherein i is an integer index variable having a property of $1 \leq i \leq max$, wherein $y_1 = a$ and $y_{max} = b$, and wherein said other numbers y_i satisfy the following properties:

$a < y_i < b$, and

$y_{i-1} < y_i$;

computer circuitry configured to generate a quantity of values v_i, wherein i is an integer index variable having a property of $1 \leq i \leq max$, wherein there are the same quantity of said values v_i as said numbers y_i, and wherein said values v_i satisfy the following properties:

$v_{i-1} < v_i$, and

min $|f(x)| \approx v_1 \leq v_{max} \approx \max |f(x)|$, wherein the operations min $|f(x)|$ and max $|f(x)|$ return the minimum and maximum absolute value of function f(x), respectively;

computer circuitry configured to generate a cubic spline g(y) with breakpoints comprising cubic spline said numbers y_i such that $g(y_i) = v_i$;

computer circuitry configured to integrate said cubic spline g(y) from a to b to obtain a value I_1;

computer circuitry configured to output said cubic spline g(y), said function f(x), and a designation of said predetermined level of accuracy;

computer circuitry configured to receive a value I_2; and

computer circuitry configured to compute said estimate using said value I_1 and said value I_2.

75. (new) A computer for use in a convolution computation of two vectors, the computer comprising:

computer circuitry configured to define a first vector M_1 and a second vector M_2, said first vector M_1 and said second vector M_2 being of equivalent size;

computer circuitry configured to define a first random vector S_1 and a second random vector S_2, said first random vector S_1 and second random vector S_2 being the same size as said first vector M_1 and said second vector M_2;

computer circuitry configured to define a first random number α, a second random number β, a third random number γ, a fourth random number β', and a fifth random number γ';

computer circuitry configured to compute a set of disguised arguments from the following matrix operations:

$$(\alpha M_1 - S_1);$$

$$(\alpha M_2 + S_2),$$

$$(\beta M_1 - \gamma S_1),$$

$$(\beta M_2 - \gamma S_2),$$

$$(\beta' M_1 - \gamma' S_1), \text{ and}$$

$$(\beta' M_2 - \gamma' S_2);$$

computer circuitry configured to output said set of disguised arguments;

computer circuitry configured to receive a matrix U, a matrix U', and a matrix W computed from said disguised arguments;

computer circuitry configured to derive a convolution of said vectors M1 and M2 using said matrix U, said matrix U', said matrix W, said first random number α, said

second random number β, said third random number γ, said fourth random number β', and said fifth random number γ'.

76. (new) A computer for use in the computation of a solution to a linear differential equation $Ly = F(x,y)$ of order N, wherein N is a positive integer, the linear differential equation having boundary conditions $y(x_i) = y_i$, wherein i is an integer index variable and $0 \leq i < N$, the computer comprising:

computer circuitry configured to define a cubic spline g(x) and a function $u(x) = Lg(x)$, wherein $Lg(x)$ is a linear differential equation of order N;

computer circuitry configured to output a set of disguised arguments comprising a set of equations as follows:

$$Ly = F(x,y) + u(x), \text{ and}$$

$$y(x_i) = y_i + u(x_i),$$

for computation of a solution function for said set of equations; and

computer circuitry configured to derive a solution to said linear differential equation Ly from said solution function.

77. (new) A computer comprising:

computer circuitry configured to automatically apply one or more transformation techniques to said symbolic mathematical expression, said one or more transformation techniques selected from the group consisting of disguising constants in said symbolic

mathematical expression and replacing variable names in said symbolic mathematical expression.

78. (new) A computer comprising:

computer circuitry configured to automatically apply one or more transformation techniques to said symbolic mathematical expression, said one or more transformation techniques selected from the group consisting of applying at least one mathematical identity function to said symbolic mathematical expression and applying at least one expansion of unity to said symbolic mathematical expression.

79. (new) A computer for use in a computation for detecting edges of an image, wherein the image is represented by an nxn array of pixel values p(x,y) between 0 and 100,000 on a unit square $0 \leq x,y \leq 1$, wherein n is a positive integer, the computer comprising:

computer circuitry configured to define at least ten ordered number pairs x_i,y_i, wherein i is an integer index variable having a property of $1 \leq i \leq max$, wherein $x_1,y_1 = 0$, wherein $x_{max},y_{max} = 1$, and wherein the remaining ordered number pairs x_i,y_i satisfy the following properties:

$$0 < x_i,y_i < 1, \text{ and}$$

$$x_{i-1},y_{i-1} < x_i,y_i;$$

computer circuitry configured to define a plurality of random values $v_{i,j}$ such that $0 \leq v_{i,j} \leq 50,000$;

computer circuitry configured to define four number pairs a_k,b_k, wherein k is an integer index variable having a property of $1 \leq k \leq 4$, each said number pair a_k,b_k comprising positive random numbers such that $a_1 = \min(a_k)$, $a_4 = \max(a_k)$, $b_1 = \min(b_k)$, and $b_4 = \max(b_k)$;

a^2

computer circuitry configured to define a bi-cubic spline s(x,y), such that $s(x_i,y_i) = v_{i,j}$;

computer circuitry configured to determine a linear change of coordinates from (x,y) coordinates to (u,v) coordinates that maps said unit square into a rectangle with vertices (a_k,b_k);

computer circuitry configured to converting said pixel values p(x,y) to pixel values p(u,v);

computer circuitry configured to converting said bi-cubic spline s(x,y) to a bi-cubic spline s(u,v);

computer circuitry configured to output said pixel values p(u,v) and said bi-cubic spline s(u,v);

computer circuitry configured to receive an image e(u,v) computed using said pixel values p(u,v) and said bi-cubic spline s(u,v); and

computer circuitry configured to derive edges of said image e(u,v).

80. (new) A computer for use in a template matching computation for image analysis, the template matching comprising computation of a score matrix $C_{I,P}$, the score matrix $C_{I,P}$

comprising an approximation of whether an image object P of size *n*x*n* appears in a larger image I of size *N*x*N*, wherein *n* and *N* are positive integers, the computer comprising:

computer circuitry configured to define a random matrix S1 of size *N*x*N*, and to define a random matrix S2 of size *n*x*n*;

computer circuitry configured to define a first random number α, a second random number β, a third random number γ, a fourth random number β', and a fifth random number γ';

computer circuitry configured to compute a set of disguised arguments comprising the following matrices:

$\alpha I + S1$,

$\alpha P + S2$,

$\beta I - \gamma S1$,

$\beta P - \gamma S2$,

$\beta' I - \gamma' S1$, and

$\beta' P - \gamma' S2$;

computer circuitry configured to output said set of disguised arguments;

computer circuitry configured to receive a matrix U, a matrix U', and a matrix W computed from said set of disguised arguments;

computer circuitry configured to derive said score matrix $C_{I,P}$ using said matrix U, said matrix U', said matrix W, said first random number α, said second random number β,

said third random number γ, said fourth random number β', and said fifth random number γ'.

81. (new) A computer for securely outsourcing the sorting of a sequence of *n* numbers E = $\{e_1,..., e_n\}$, wherein *n* is a positive integer, the computer comprising:

computer circuitry configured to define a strictly increasing function f();

computer circuitry configured to define a random sorted sequence $\Lambda = \{\lambda_1,...,\lambda_n\}$ of *n* numbers;

computer circuitry configured to compute sequence E' = f(E) and sequence Λ' = $f(\Lambda)$, wherein f(E) is a sequence obtained from E by replacing every element e_i by $f(e_i)$;

computer circuitry configured to compute set W as follows: $W = E' \cup \Lambda'$;

computer circuitry configured to randomly permute set W with said first computer;

computer circuitry configured to output said randomly permuted set W;

computer circuitry configured to receive sorted set W', said sorted set W' derived from said randomly permuted set W;

computer circuitry configured to derive sorted sequence E from said sorted set W'.

82. (new) A computer for use in a text string pattern matching computation, wherein T is a text string of length *N*, wherein *N* is a positive integer, said text string T comprising *N* text symbols, and wherein P is a text pattern of length *n*, wherein *n* is a positive integer that is smaller

than *N*, said text pattern P comprising *n* text symbols, and wherein alphabet A comprises the plurality of possible text symbols that could appear in text string T or text pattern P, the computer comprising:

computer circuitry configured to select a text symbol from alphabet A;

computer circuitry configured to replace each instance of said selected text symbol in said text string T with the number 1, and replacing each other text symbol in text string T with the number 0, the resultant text string being designated T_x;

computer circuitry configured to replace each instance of said selected text symbol in said text pattern P with the number 1, and replacing each other text symbol in text pattern P with the number 0, the resultant text pattern being designated P_x;

computer circuitry configured to augment said text pattern P_x into a text string P' of length by adding zeros thereto;

computer circuitry configured to output said text string P' and said text string T_x;

computer circuitry configured to receive a value D_x computed using said text string P' and said text string T_x as follows:

$$D_x(i) = \sum_{k=0}^{n-1} T_x(i+k)P'(k), \qquad 0 \le i \le N-n; \text{ and}$$

computer circuitry configured to compute a score matrix $C_{T,P}$ using said values D_x. for all said text symbols in said alphabet A.